

Package ‘peppwR’

May 9, 2026

Title Power Analysis for Phosphopeptide Abundance Hypothesis Test

Version 0.1.0

Author Dan MacLean [aut, cre]

Maintainer Dan MacLean <dan.maclean@tsl.ac.uk>

Description Estimate best fit distributions and do power analysis for hypothesis tests on phosphopeptide abundance data.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

URL <https://teammaclean.github.io/peppwR/>,
<https://github.com/TeamMacLean/peppwR>

BugReports <https://github.com/TeamMacLean/peppwR/issues>

Depends R (>= 4.1.0)

Imports cowplot, dplyr, fitdistrplus, ggplot2, methods, purrr,
RColorBrewer, scales, tibble, tidyr, univariateML

Suggests bench, knitr, pkgdown, rmarkdown, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2026-03-30 09:20:08 UTC

Contents

compute_dataset_mnar	2
compute_missingness	3
fit_distributions	4
get_distribution_preset	5
is_count_data	6

new_peppwr_fits	6
new_peppwr_power	7
plot.peppwr_fits	7
plot.peppwr_power	8
plot_density_overlay	8
plot_missingness	9
plot_param_distribution	10
plot_power_heatmap	10
plot_power_vs_effect	11
plot_qq	12
power_analysis	12
power_analysis.character	13
power_analysis.peppwr_fits	14
print.peppwr_fits	15
print.peppwr_power	15
print.summary.peppwr_fits	16
print.summary.peppwr_power	16
run_power_sim	17
run_power_sim_empirical	18
run_power_sim_fdr	18
run_power_sim_with_missingness	19
simulate_empirical	20
simulate_experiment	21
simulate_with_missingness	21
summary.peppwr_fits	22
summary.peppwr_power	23
test_bayes_t	23
test_bootstrap_t	24
test_rankprod	24
test_wilcoxon	25
validate_peppwr_fits	25
validate_peppwr_power	26

Index	27
--------------	-----------

compute_dataset_mnar *Compute dataset-level MNAR metric*

Description

Calculates Spearman correlation between $\log(\text{mean_abundance})$ and NA rate across peptides to detect whether low-abundance peptides have more missing values than high-abundance peptides.

Usage

```
compute_dataset_mnar(missingness)
```

Arguments

missingness A tibble with columns na_rate and mean_abundance (as produced by fit_distributions)

Value

A list with:

- correlation: Spearman correlation coefficient (negative = MNAR pattern)
- p_value: p-value for the correlation
- n_peptides: Number of peptides with missing data used in calculation
- interpretation: Human-readable interpretation string

MNAR Detection

MNAR (Missing Not At Random) in mass spectrometry typically manifests as low-abundance peptides having higher rates of missing values due to detection limits. This function detects this pattern by correlating mean abundance with NA rate across all peptides.

A negative correlation indicates that low-abundance peptides have more missing values - the hallmark of detection-limit-driven MNAR.

Interpretation

Correlation (r)	Interpretation
$r > -0.1$	No evidence of MNAR
$-0.3 < r < -0.1$	Weak evidence
$-0.5 < r < -0.3$	Moderate evidence
$r < -0.5$	Strong evidence of MNAR

compute_missingness *Compute missingness statistics for a vector of values*

Description

Calculates the number and proportion of missing (NA) values.

Usage

```
compute_missingness(values)
```

Arguments

values Numeric vector (may contain NAs)

Value

List with:

- `n_total`: Total number of values
- `n_missing`: Number of NA values
- `na_rate`: Proportion missing (0-1)

See Also

[compute_dataset_mnar\(\)](#) for dataset-level MNAR detection

`fit_distributions` *Fit distributions to peptide abundance data*

Description

Fits candidate distributions to each peptide's abundance values and selects the best fit by AIC. Also computes missingness statistics including dataset-level MNAR detection.

Usage

```
fit_distributions(data, id, group, value, distributions = "continuous")
```

Arguments

<code>data</code>	A data frame containing peptide abundance data
<code>id</code>	Column name for peptide identifier
<code>group</code>	Column name for group/condition
<code>value</code>	Column name for abundance values
<code>distributions</code>	Which distributions to fit: "continuous" (default), "counts", "all", or a character vector of distribution names

Value

A `peppwr_fits` object containing:

- `$data`: Nested tibble with original data and fit results
- `$best`: Best-fitting distribution for each peptide
- `$missingness`: Per-peptide missingness statistics
- `$dataset_mnar`: Dataset-level MNAR correlation and interpretation

Missingness Tracking

The returned object includes:

- Per-peptide NA rates (in `$missingness`)
- Dataset-level MNAR correlation (in `$dataset_mnar`)

The dataset-level MNAR metric correlates $\log(\text{mean_abundance})$ with NA rate across peptides. A negative correlation indicates low-abundance peptides have more missing values - typical of detection-limit-driven MNAR.

Print the result to see both metrics. For small sample sizes ($N < 15$), the dataset-level correlation is more reliable than per-peptide scores.

See Also

[compute_dataset_mnar\(\)](#) for dataset-level MNAR details, [plot_missingness\(\)](#) to visualize missingness patterns

get_distribution_preset

Get distributions for a preset

Description

Get distributions for a preset

Usage

```
get_distribution_preset(preset = "continuous")
```

Arguments

preset One of "continuous" (default), "counts", or "all"

Value

Character vector of distribution names

is_count_data	<i>Check if data appears to be count data (non-negative integers)</i>
---------------	---

Description

Check if data appears to be count data (non-negative integers)

Usage

```
is_count_data(x)
```

Arguments

x	Numeric vector
---	----------------

Value

TRUE if data looks like counts, FALSE otherwise

new_peppwr_fits	<i>Create a new peppwr_fits object</i>
-----------------	--

Description

Create a new peppwr_fits object

Usage

```
new_peppwr_fits(
  data,
  fits,
  best,
  call,
  missingness = NULL,
  dataset_mnar = NULL
)
```

Arguments

data	Original data (nested tibble)
fits	Fit results per peptide (list of tibbles with dist, loglik, aic)
best	Best-fitting distribution per peptide (character vector)
call	Original function call
missingness	Tibble with missingness statistics per peptide (optional)
dataset_mnar	Dataset-level MNAR metric (optional, from compute_dataset_mnar)

Value

A peppwr_fits object

new_peppwr_power	<i>Create a new peppwr_power object</i>
------------------	---

Description

Create a new peppwr_power object

Usage

```
new_peppwr_power(mode, question, answer, simulations, params, call)
```

Arguments

mode	Either "aggregate" or "per_peptide"
question	What was solved for: "power", "sample_size", or "effect_size"
answer	The computed answer
simulations	List of simulation details
params	List of input parameters used
call	Original function call

Value

A peppwr_power object

plot.peppwr_fits	<i>Plot method for peppwr_fits</i>
------------------	------------------------------------

Description

Creates a bar chart showing the count of best-fit distributions

Usage

```
## S3 method for class 'peppwr_fits'
plot(x, ...)
```

Arguments

x	A peppwr_fits object
...	Additional arguments (ignored)

Value

A ggplot object

plot.peppwr_power *Plot method for peppwr_power*

Description

Creates power curves or % peptides at threshold plots

Usage

```
## S3 method for class 'peppwr_power'  
plot(x, ...)
```

Arguments

x A peppwr_power object
... Additional arguments (ignored)

Value

A ggplot object

plot_density_overlay *Plot density overlay: observed histogram with fitted density curve*

Description

Plot density overlay: observed histogram with fitted density curve

Usage

```
plot_density_overlay(fits, peptide_id = NULL, n_overlay = 6)
```

Arguments

fits A peppwr_fits object
peptide_id Specific peptide to plot (NULL for multiple)
n_overlay Number of peptides to overlay when peptide_id is NULL

Value

A ggplot object

plot_missingness	<i>Plot missingness statistics</i>
------------------	------------------------------------

Description

Creates a visualization of missing data patterns with two panels:

Usage

```
plot_missingness(fits)
```

Arguments

`fits` A `peppwr_fits` object

Details

- **Panel 1:** Distribution of NA rates across peptides
- **Panel 2:** Mean abundance vs NA rate scatter plot showing the **dataset-level MNAR correlation**. The subtitle displays the Spearman correlation coefficient and p-value. A negative correlation indicates that low-abundance peptides have more missing values - the hallmark of detection-limit-driven MNAR.

Value

A ggplot object or gtable (combined panels)

MNAR Detection

MNAR (Missing Not At Random) in mass spectrometry typically manifests as low-abundance peptides having higher rates of missing values due to detection limits. Panel 2 visualizes this relationship and reports the correlation coefficient.

See Also

[compute_dataset_mnar\(\)](#) for the underlying correlation calculation

`plot_param_distribution`*Plot distribution of fitted parameters across peptidome*

Description

Plot distribution of fitted parameters across peptidome

Usage

```
plot_param_distribution(fits)
```

Arguments

`fits` A `peppwr_fits` object

Value

A `ggplot` object

`plot_power_heatmap`*Plot power heatmap: N x effect size grid*

Description

Plot power heatmap: N x effect size grid

Usage

```
plot_power_heatmap(  
  distribution,  
  params,  
  n_range,  
  effect_range,  
  n_steps = 6,  
  n_sim = 100,  
  test = "wilcoxon",  
  alpha = 0.05  
)
```

Arguments

distribution	Distribution name
params	Distribution parameters
n_range	Range of sample sizes (vector of length 2)
effect_range	Range of effect sizes (vector of length 2)
n_steps	Number of grid points per dimension
n_sim	Number of simulations per grid cell
test	Statistical test to use
alpha	Significance level

Value

A ggplot object

`plot_power_vs_effect` *Plot power vs effect size at fixed N*

Description

Plot power vs effect size at fixed N

Usage

```
plot_power_vs_effect(power_result, effect_range, n_steps = 10, n_sim = NULL)
```

Arguments

power_result	A peppwr_power object
effect_range	Range of effect sizes to explore
n_steps	Number of effect size values to compute
n_sim	Number of simulations per point

Value

A ggplot object

plot_qq	<i>Plot QQ plots for goodness-of-fit</i>
---------	--

Description

Plot QQ plots for goodness-of-fit

Usage

```
plot_qq(fits, peptide_id = NULL, n_plots = 6)
```

Arguments

fits	A peppwr_fits object
peptide_id	Specific peptide to plot (NULL for multiple)
n_plots	Number of peptides to plot when peptide_id is NULL

Value

A ggplot object

power_analysis	<i>Power analysis for peptide abundance data</i>
----------------	--

Description

Power analysis for peptide abundance data

Usage

```
power_analysis(distribution, ...)
```

Arguments

distribution	Distribution name (character) or peppwr_fits object for per-peptide mode
...	Additional arguments

Value

A peppwr_power object

 power_analysis.character

Power analysis with specified distribution (aggregate mode)

Description

Power analysis with specified distribution (aggregate mode)

Default power analysis method (aggregate mode with defaults)

Usage

```
## S3 method for class 'character'
power_analysis(
  distribution,
  params,
  effect_size = NULL,
  n_per_group = NULL,
  target_power = NULL,
  alpha = 0.05,
  test = "wilcoxon",
  find = "power",
  n_sim = 1000,
  ...
)

## Default S3 method:
power_analysis(distribution, ...)
```

Arguments

distribution	Distribution name (e.g., "norm", "gamma", "lnorm")
params	List of distribution parameters
effect_size	Fold change to detect
n_per_group	Sample size per group (required for find="power")
target_power	Target power (required for find="sample_size" or find="effect_size")
alpha	Significance level (default 0.05)
test	Statistical test to use (default "wilcoxon")
find	What to solve for: "power", "sample_size", or "effect_size"
n_sim	Number of simulations (default 1000)
...	Additional arguments (ignored)

Value

A peppwr_power object

power_analysis.peppwr_fits

Power analysis for per-peptide mode using fitted distributions

Description

Power analysis for per-peptide mode using fitted distributions

Usage

```
## S3 method for class 'peppwr_fits'
power_analysis(
  distribution,
  effect_size = NULL,
  n_per_group = NULL,
  target_power = NULL,
  alpha = 0.05,
  test = "wilcoxon",
  find = "power",
  n_sim = 1000,
  on_fit_failure = "exclude",
  proportion_threshold = 0.5,
  include_missingness = FALSE,
  apply_fdr = FALSE,
  prop_null = 0.9,
  fdr_threshold = 0.05,
  ...
)
```

Arguments

distribution	A peppwr_fits object from fit_distributions()
effect_size	Fold change to detect
n_per_group	Sample size per group (required for find="power")
target_power	Target power (required for find="sample_size")
alpha	Significance level (default 0.05)
test	Statistical test to use (default "wilcoxon")
find	What to solve for: "power" or "sample_size"
n_sim	Number of simulations per peptide (default 1000)
on_fit_failure	How to handle failed fits: "exclude", "empirical", or "lognormal"
proportion_threshold	Proportion of peptides that must reach target_power (default 0.5)
include_missingness	If TRUE, incorporate peptide-specific NA rates into simulations

apply_fdr	If TRUE, use FDR-aware simulation with Benjamini-Hochberg correction. Note: not compatible with test = "bayes_t" (Bayes factors cannot be converted to p-values)
prop_null	Proportion of true null peptides (default 0.9 = 90% unchanged)
fdr_threshold	FDR threshold for calling discoveries (default 0.05)
...	Additional arguments (ignored)

Value

A peppwr_power object

print.peppwr_fits *Print method for peppwr_fits*

Description

Print method for peppwr_fits

Usage

```
## S3 method for class 'peppwr_fits'
print(x, ...)
```

Arguments

x	A peppwr_fits object
...	Additional arguments (ignored)

Value

The object invisibly

print.peppwr_power *Print method for peppwr_power*

Description

Print method for peppwr_power

Usage

```
## S3 method for class 'peppwr_power'
print(x, ...)
```

Arguments

x A peppwr_power object
... Additional arguments (ignored)

Value

The object invisibly

`print.summary.peppwr_fits`

Print method for summary.peppwr_fits

Description

Print method for `summary.peppwr_fits`

Usage

```
## S3 method for class 'summary.peppwr_fits'  
print(x, ...)
```

Arguments

x A `summary.peppwr_fits` object
... Additional arguments (ignored)

Value

The object invisibly

`print.summary.peppwr_power`

Print method for summary.peppwr_power

Description

Print method for `summary.peppwr_power`

Usage

```
## S3 method for class 'summary.peppwr_power'  
print(x, ...)
```

Arguments

x A summary.peppwr_power object
 ... Additional arguments (ignored)

Value

The object invisibly

run_power_sim	<i>Run power simulation</i>
---------------	-----------------------------

Description

Run power simulation

Usage

```
run_power_sim(
  distribution,
  params,
  n_per_group,
  effect_size,
  alpha = 0.05,
  test = "wilcoxon",
  n_sim = 1000
)
```

Arguments

distribution	Distribution name
params	List of distribution parameters
n_per_group	Number of samples per group
effect_size	Fold change for treatment
alpha	Significance level
test	Statistical test to use ("wilcoxon", "bootstrap_t", "bayes_t", "rankprod")
n_sim	Number of simulations

Value

Power estimate (proportion of significant results)

`run_power_sim_empirical`*Run power simulation using empirical bootstrap*

Description

Estimates power by repeatedly resampling from observed data rather than sampling from fitted distributions.

Usage

```
run_power_sim_empirical(  
  raw_data,  
  n_per_group,  
  effect_size,  
  alpha = 0.05,  
  test = "wilcoxon",  
  n_sim = 1000  
)
```

Arguments

<code>raw_data</code>	Numeric vector of observed values
<code>n_per_group</code>	Number of samples per group
<code>effect_size</code>	Fold change for treatment
<code>alpha</code>	Significance level
<code>test</code>	Statistical test to use
<code>n_sim</code>	Number of simulations

Value

Power estimate (proportion of significant results)

`run_power_sim_fdr`*Run FDR-aware power simulation for whole peptidome*

Description

Simulates an entire peptidome experiment with a mix of true nulls and true alternatives, then applies Benjamini-Hochberg correction to compute FDR-adjusted power.

Usage

```
run_power_sim_fdr(  
  fits,  
  effect_size,  
  n_per_group,  
  prop_null = 0.9,  
  fdr_threshold = 0.05,  
  alpha = 0.05,  
  test = "wilcoxon",  
  n_sim = 1000  
)
```

Arguments

fits	A peppwr_fits object
effect_size	Fold change for treatment
n_per_group	Number of samples per group
prop_null	Proportion of peptides that are true nulls (no effect). Default 0.9 (90% unchanged).
fdr_threshold	FDR threshold for calling discoveries. Default 0.05.
alpha	Nominal significance level (used for simulation). Default 0.05.
test	Statistical test to use. Default "wilcoxon".
n_sim	Number of simulation iterations. Default 1000.

Value

FDR-adjusted power estimate (proportion of true alternatives detected)

run_power_sim_with_missingness
Run power simulation with missingness

Description

Estimates power accounting for realistic missing data patterns.

Usage

```
run_power_sim_with_missingness(  
  distribution,  
  params,  
  n_per_group,  
  effect_size,  
  na_rate = 0,  
  mmar_score = 0,
```

```

    alpha = 0.05,
    test = "wilcoxon",
    n_sim = 1000
  )

```

Arguments

distribution	Distribution name
params	List of distribution parameters
n_per_group	Number of samples per group
effect_size	Fold change for treatment
na_rate	Proportion of values that will be NA
mnar_score	MNAR intensity (0 = MCAR)
alpha	Significance level
test	Statistical test to use
n_sim	Number of simulations

Value

Power estimate (proportion of significant results)

simulate_empirical *Simulate experiment using empirical bootstrap*

Description

Resamples from observed data instead of using parametric distributions. Useful when distribution fitting fails or for non-parametric analysis.

Usage

```
simulate_empirical(raw_data, n_per_group, effect_size)
```

Arguments

raw_data	Numeric vector of observed values
n_per_group	Number of samples per group
effect_size	Fold change for treatment (multiplicative effect)

Value

List with control and treatment vectors

simulate_experiment *Simulate an experiment with control and treatment groups*

Description

Simulate an experiment with control and treatment groups

Usage

```
simulate_experiment(distribution, params, n_per_group, effect_size)
```

Arguments

distribution	Distribution name (e.g., "norm", "gamma", "lnorm")
params	List of distribution parameters
n_per_group	Number of samples per group
effect_size	Fold change for treatment (multiplicative effect)

Value

List with control and treatment vectors

simulate_with_missingness
Simulate experiment with realistic missingness

Description

Generates control and treatment samples from a distribution, then introduces missing values according to the specified rate and MNAR pattern.

Usage

```
simulate_with_missingness(  
  distribution,  
  params,  
  n_per_group,  
  effect_size,  
  na_rate = 0,  
  mnar_score = 0  
)
```

Arguments

distribution	Distribution name (e.g., "norm", "gamma", "lnorm")
params	List of distribution parameters
n_per_group	Number of samples per group
effect_size	Fold change for treatment (multiplicative effect)
na_rate	Proportion of values to make NA (0-1)
mnar_score	MNAR intensity: 0 = MCAR, positive = low values more likely to be missing. Typical values: 0-3.

Value

List with control and treatment vectors (may contain NAs)

summary.peppwr_fits *Summary method for peppwr_fits*

Description

Summary method for peppwr_fits

Usage

```
## S3 method for class 'peppwr_fits'
summary(object, ...)
```

Arguments

object	A peppwr_fits object
...	Additional arguments (ignored)

Value

A list with summary statistics

summary.peppwr_power *Summary method for peppwr_power*

Description

Summary method for peppwr_power

Usage

```
## S3 method for class 'peppwr_power'
summary(object, ...)
```

Arguments

object	A peppwr_power object
...	Additional arguments (ignored)

Value

A list with summary statistics

test_bayes_t *Bayes factor t-test*

Description

Computes Bayes factor for difference between two groups

Usage

```
test_bayes_t(control, treatment)
```

Arguments

control	Control group values
treatment	Treatment group values

Value

Bayes factor (BF10: evidence for alternative over null)

test_bootstrap_t	<i>Bootstrap-t test</i>
------------------	-------------------------

Description

Performs a bootstrap-t test comparing two groups

Usage

```
test_bootstrap_t(control, treatment, n_boot = 1000)
```

Arguments

control	Control group values
treatment	Treatment group values
n_boot	Number of bootstrap iterations (default 1000)

Value

p-value from the test

test_rankprod	<i>Rank Products test</i>
---------------	---------------------------

Description

Performs rank products test comparing two groups Simplified implementation for two-group comparison

Usage

```
test_rankprod(control, treatment, n_perm = 1000)
```

Arguments

control	Control group values
treatment	Treatment group values
n_perm	Number of permutations for p-value estimation (default 1000)

Value

p-value from the test

test_wilcoxon	<i>Wilcoxon rank-sum test</i>
---------------	-------------------------------

Description

Wilcoxon rank-sum test

Usage

```
test_wilcoxon(control, treatment)
```

Arguments

control	Control group values
treatment	Treatment group values

Value

p-value from the test

validate_peppwr_fits	<i>Validate a peppwr_fits object</i>
----------------------	--------------------------------------

Description

Validate a peppwr_fits object

Usage

```
validate_peppwr_fits(x)
```

Arguments

x	Object to validate
---	--------------------

Value

The validated object (invisibly)

validate_peppwr_power *Validate a peppwr_power object*

Description

Validate a peppwr_power object

Usage

```
validate_peppwr_power(x)
```

Arguments

x Object to validate

Value

The validated object (invisibly)

Index

`compute_dataset_mnar`, 2
`compute_dataset_mnar()`, 4, 5, 9
`compute_missingness`, 3

`fit_distributions`, 4

`get_distribution_preset`, 5

`is_count_data`, 6

`new_peppwr_fits`, 6
`new_peppwr_power`, 7

`plot.peppwr_fits`, 7
`plot.peppwr_power`, 8
`plot_density_overlay`, 8
`plot_missingness`, 9
`plot_missingness()`, 5
`plot_param_distribution`, 10
`plot_power_heatmap`, 10
`plot_power_vs_effect`, 11
`plot_qq`, 12
`power_analysis`, 12
`power_analysis.character`, 13
`power_analysis.default`
 (`power_analysis.character`), 13
`power_analysis.peppwr_fits`, 14
`print.peppwr_fits`, 15
`print.peppwr_power`, 15
`print.summary.peppwr_fits`, 16
`print.summary.peppwr_power`, 16

`run_power_sim`, 17
`run_power_sim_empirical`, 18
`run_power_sim_fdr`, 18
`run_power_sim_with_missingness`, 19

`simulate_empirical`, 20
`simulate_experiment`, 21
`simulate_with_missingness`, 21
`summary.peppwr_fits`, 22

`summary.peppwr_power`, 23

`test_bayes_t`, 23
`test_bootstrap_t`, 24
`test_rankprod`, 24
`test_wilcoxon`, 25

`validate_peppwr_fits`, 25
`validate_peppwr_power`, 26